

Sun Expert Exchange

Technical Knowledge Base for Sun Inner Circle Members



Fast Track to Solaris 10 Adoption: Dynamic Tracing A Sun Expert Exchange Discussion

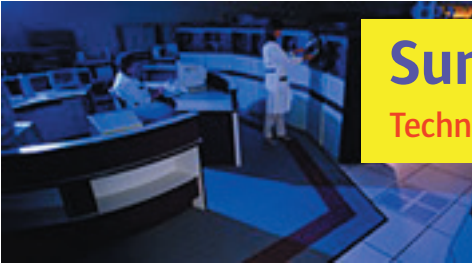
DTrace, one of the powerful new features of the Solaris 10 OS, is the comprehensive dynamic tracing framework for troubleshooting system problems in real time and tuning applications for performance, all with little or no performance impact.

This summary includes highlights of the hour-long Q&A,* organized into the following sections:

- General Information Pages 2-4
- Documentation & Training Page 5
- Performance Issues Page 6
- Installation & Configuration Pages 7-8
- Compatibility Issues Page 9
- Functionality & Usability Issues Pages 10-13

In addition to questions and answers, you'll also find references and links to additional resources provided by Sun.

*Note: The information contained in this transcript, taken directly from a live Sun Expert Exchange event, has been edited for clarity and adherence to trademark guidelines.



Sun Expert Exchange

Technical Knowledge Base for Sun Inner Circle Members



Fast Track to Solaris 10 Adoption: Dynamic Tracing General Information

1. What is DTrace?
2. What will DTrace allow me to do?
3. How about a simple scenario of how DTrace would be used?
4. Would you give an example of a real world problem that you solved with DTrace?
5. Do I have to pay extra for DTrace?
6. Is DTrace only available in the Solaris 10 OS?
7. Is there a release date yet for the Solaris 10 OS?
8. Can you supply a mini product roadmap for DTrace — convenience features/addons, integration with SMC, etc.?
9. Can DTrace be used for all Solaris OS versions from 2.5.1 to 10?
10. Do you have to have a deep understanding of the kernel for DTrace to be useful?
11. Will we ever hear the back-story on DTrace, like how you guys worked up the nerve to dynamically instrument arbitrary running pieces of the system? How hard was it to apply aspects to the Solaris OS, etc.?

Q: What is DTrace?

A: Imagine if any question you had about your systems could be magically answered — instantly. Imagine how much easier it would be to find system bottlenecks and understand complicated performance issues. That's the dramatic effect dynamic tracing, or DTrace, a comprehensive dynamic tracing framework for the Solaris OS, can have on your data center.

More powerful than any other tool available, DTrace is an unmatched dynamic tracing framework for troubleshooting your network and tuning system performance in real time. DTrace lets you see your entire Solaris OS system in an entirely new way, revealing systemic problems that were previously invisible and fixing performance issues that used to go unresolved.

Q: What will DTrace allow me to do?

A: Here are some example uses:

- Examine the behavior of user programs and the Solaris OS and quickly identify the root causes of system and application bottlenecks
- Highlight trends and patterns to tune systems for best performance
- Track down performance problems across many layers of software
- Locate the cause of aberrant behavior
- Write reusable scripts for common or complex routines
- Specify the data DTrace collects, the actions it takes, and the conditions under which it should take those actions

Q: How about a simple scenario of how DTrace would be used?

A: DTrace will find a place in the heart of almost every even vaguely serious Solaris OS user.



Sun Expert Exchange

Technical Knowledge Base for Sun Inner Circle Members



Personally, I find myself using it every day in various aspects of my job — often in situations that would have been impossible or incredible arduous without DTrace.

As for some simple examples, let's say you're a developer working on a new application. Often you want to know the value of a variable each time a function is called or at a particular point in a function. A debugger would be a pain because you'd have to stop, print the value, and continue a bazillion times. Recompiling the app to add a `printf(3C)` is a pain because it takes time. Instead, you can use DTrace to instrument exactly the spot you want and trace exactly the data you want.

Now, let's say you're a systems administrator and you want to find out what activity on your system is generating I/O. Without DTrace it's a serious hassle at best; with DTrace you can just enable a couple of probes and get your answer. Check out the I/O chapter in the Solaris Dynamic Tracing Guide on DTrace's BigAdmin website for this particular example.

Q: Would you give an example of a real world problem that you solved with DTrace?

A: You bet. Check out [this paper](#).

Q: Do I have to pay extra for DTrace?

A: DTrace is bundled with the Solaris 10 OS at no extra charge.

Q: Is DTrace only available in the Solaris 10 OS?

A: DTrace is only in the Solaris 10 OS but some folks have success moving an app to a Solaris 10 system, tuning and debugging it there, and returning it to the Solaris 9 or 8 operating systems.

Q: Is there a release date yet for the Solaris 10 OS?

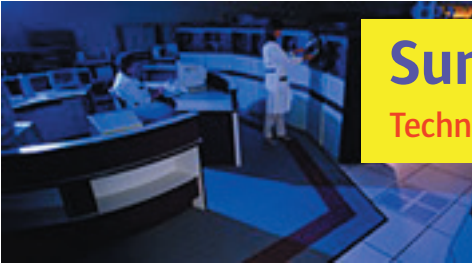
A: The Solaris 10 OS is scheduled to ship at the end of 2004. Early access to the Solaris 10 OS can be achieved today by joining the [Software Express program](#).

Q: Can you supply a mini product roadmap for DTrace — convenience features/addons, integration with SMC, etc.?

A: The immediate future for DTrace is kernel-level providers that export stable semantics. You're already seeing this with the new `sched` and `proc` providers in Solaris Express 6/04, and we have several more on the way in this department. The next direction is to allow the same kind of stable semantics for user-level tracing. Right now, you can instrument any instruction (and/or any function) in any running process — but that doesn't necessarily help you if you want to, say, time a transaction. To solve this, we want to allow programs to export their own stable probes with well-defined semantics. This work is currently underway; stay tuned!

Q: Can DTrace be used for all Solaris OS versions from 2.5.1 to 10?

A: DTrace is only available on the Solaris 10 OS. We have many customers who install the Solaris 10 OS on a test system and then bring their apps over so they can look at them with DTrace. After the application's behavior has been examined and any issues rectified, the app can then be moved back to a non-Solaris 10 OS. Get the Solaris 10 OS through Software Express and you can use



Sun Expert Exchange

Technical Knowledge Base for Sun Inner Circle Members



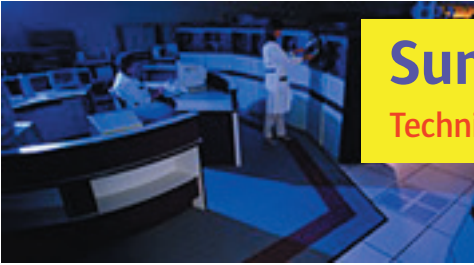
it to examine any Solaris application.

Q: Do you have to have a deep understanding of the kernel for DTrace to be useful?

A: Absolutely not. We have developed several providers that abstract away kernel implementation details. For example, the “sched” provider makes available probes related to CPU scheduling, like “on-CPU,” “off-CPU,” “enqueue,” “dequeue,” etc. You don’t need to understand the implementation details of the scheduler to use it. See the DTrace documentation for details.

Q: Will we ever hear the back-story on DTrace, like how you guys worked up the nerve to dynamically instrument arbitrary running pieces of the system? How hard was it to apply aspects to the Solaris OS, etc.?

A: Well, no one has ever accused us of not having nerve. This is something we’ve wanted to do for a long, long time. We had a pretty clear vision of what we wanted to do, to the point where we would say things like “damn, I needed DTrace today” — in 1998. In fact, I think our colleagues got a little sick of “You know, DTrace will solve that problem” — when we didn’t have so much as a line of code written.



Sun Expert Exchange

Technical Knowledge Base for Sun Inner Circle Members



Fast Track to Solaris 10 Adoption: Dynamic Tracing Documentation & Training

1. Are there any example scripts or tutorials available? Where can I find DTrace documentation for the details?
2. There are some examples linked off <http://www.sun.com/bigadmin/content/dtrace/> but I'd like more examples. Do you plan to provide any in the near future?
3. I think we're all impressed by the "what" of DTrace. What we really need are good examples of timing application functions and calls, and so on.
4. Do you know in which of the Solaris OS system administration training courses DTrace will be covered?
5. Are there any plans to provide a set of pre-prepared DTrace tools?

Q: Are there any example scripts or tutorials available? Where can I find DTrace documentation for the details?

A: Look for sample scripts and other documentation at www.sun.com/bigadmin

Q: There are some examples linked off <http://www.sun.com/bigadmin/content/dtrace/> but I'd like more examples. Do you plan to provide any in the near future?

A: There are tons of examples in the DTrace AnswerBook guide, available from the BigAdmin site. And there are a bunch more on the DTrace forum. And yes, we've got lots more examples planned; you can never have too many examples!

Q: I think we're all impressed by the "what" of DTrace. What we really need are good examples of timing application functions and calls, and so on.

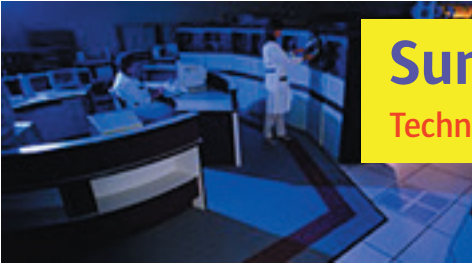
A: Go to the AnswerBook resource center as your first source for real-world examples. The documentation was 100 percent written by the three of us, and we've designed it to be as example-packed as possible. That said, there is always a need for more examples! So if there is a specific class of examples that we're missing, please let us know.

Q: Do you know in which of the Solaris OS system administration training courses DTrace will be covered?

A: I don't know the specific course number(s), but a SunEd DTrace course is in the works and will be available by the time the Solaris 10 OS GAs. Stay tuned!

Q: Are there any plans to provide a set of pre-prepared DTrace tools?

A: Actually, several existing utilities use DTrace as a backend. For example, lockstat(1M) and the new intrstat(1M). If you're asking about pre-prepared DTrace scripts, we have a ton of examples in the documentation and on the BigAdmin site — and more always on the way!



Sun Expert Exchange

Technical Knowledge Base for Sun Inner Circle Members



Fast Track to Solaris 10 Adoption: Dynamic Tracing Performance Issues

1. Doesn't DTrace work only on the kernel?
2. Does DTrace introduce any performance hits?
3. How much CPU/memory does DTrace take up while debugging kernel/system calls?
4. Could DTrace be compared to the very popular SE Toolkit that is used to gather performance data? From what I have read, it seems to be much more powerful, but I was interested in a replacement for the SE Toolkit.
5. I have noticed that when the DNLC cache is getting full, the system time is increasing. Can DTrace verify that the CPU time is spent on traversing and processing the DNLC cache? In general, can DTrace see how much time is spent on processing kernel data structure?

Q: Doesn't DTrace work only on the kernel?

A: No, DTrace works on the kernel and on applications.

Q: Does DTrace introduce any performance hits?

A: When DTrace is disabled, there is no performance effect at all. The DTrace framework allows you to enable any of 30,000 or more probes. If you just enable a few, the hit will be relatively small; if you enable them all, it will create a noticeable impact, but the machine will still be fine.

Q: How much CPU/memory does DTrace take up while debugging kernel/system calls?

A: With DTrace, you pay for what you ask. If you enable a few probes, the performance hit is negligible; if you enable more, you might start to notice it. In terms of memory, the size of DTrace's internal buffers is completely configurable (see the Buffers and Buffering and Options chapters in the Solaris Dynamic Tracing Guide). In general, the memory footprint isn't a factor on anything but the smallest systems (like 128M).

Q: Could DTrace be compared to the very popular SE Toolkit that is used to gather performance data? From what I have read, it seems to be much more powerful, but I was interested in a replacement for the SE Toolkit.

A: It's not a replacement really — it's more of a complementary technology. Let's say that you see something aberrant with the SE Toolkit. DTrace allows you to dive down on that aberration to root-cause it. If you look at the DTrace forum, this question has also been asked over there. My answer there might be slightly less rushed.

Q: I have noticed that when the DNLC cache is getting full, the system time is increasing. Can DTrace verify that the CPU time is spent on traversing and processing the DNLC cache? In general, can DTrace see how much time is spent on processing kernel data structure?

A: Absolutely. Although the system time may well be increasing simply because your miss rate is rising in the DNLC. Check the "dnlcstats" kstat. DTrace can absolutely be used to investigate this. For starters, look at using the "profile" provider — see the profile chapter of the AnswerBook resource guide.



Sun Expert Exchange

Technical Knowledge Base for Sun Inner Circle Members



Fast Track to Solaris 10 Adoption: Dynamic Tracing Installation & Configuration

1. Do I need to recompile my apps to use DTrace?
2. I just installed the Solaris 10 OS on a system and DTrace -l lists no probes. Shouldn't there be a basic set of internal probes provided?
3. Can I take it as a more powerful debugger?
4. We just ordered a Sun 8000 box with the Solaris 9 OS. Is release 10 available, and how do I get it installed? The VAR has not built the server yet.
5. Is DTrace in /usr/openwin/bin file system?
6. Is there a C API so that DTrace could be incorporated into an application for automatic performance monitoring and resolution?
7. Is it possible to have a dual-boot box with both Solaris 9 and Solaris 10 operating systems on it? If so, how can this be done, and what are the requirements?
8. I just installed the Solaris 10 OS on a system (Ultra 5) and issued the DTrace -l command. The system is running extremely slowly now and appears to be cranking away on something, even after Ctrl-c to terminate the DTrace -l call. Did I do something wrong?

Q: Do I need to recompile my apps to use DTrace?

A: Absolutely not — you don't even need to restart them.

Q: I just installed the Solaris 10 OS on a system and DTrace -l lists no probes. Shouldn't there be a basic set of internal probes provided?

A: You're probably not running as root. The fact that you're getting no output (and not an error message) is a bug that we fixed recently.

Q: Can I take it as a more powerful debugger?

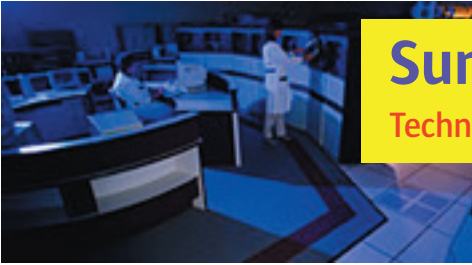
A: Yes. A much, much, much more powerful debugger!

Q: We just ordered a Sun 8000 box with the Solaris 9 OS. Is release 10 available, and how do I get it installed? The VAR has not built the server yet.

A: The Solaris 10 OS is currently available for download through the Software Express program. This is an early access release and is not recommended for use on production systems; however, we do have some customers running the Solaris 10 OS in production environments, and limited support is available. For more information, please go to <http://www.sun.com/solaris/10>

Q: Is DTrace in /usr/openwin/bin file system?

A: The DTrace(1M) utility can be found at /usr/sbin/DTrace.



Sun Expert Exchange

Technical Knowledge Base for Sun Inner Circle Members



Q: Is there a C API so that DTrace could be incorporated into an application for automatic performance monitoring and resolution?

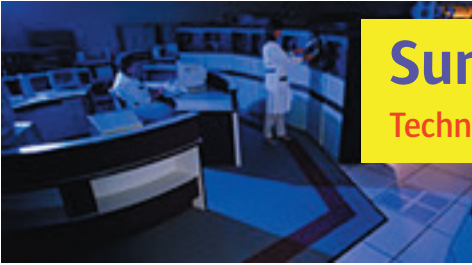
A: There isn't a publicly available API for libdtrace.so. You can write DTrace scripts and .d files, which can encapsulate your performance monitoring knowledge.

Q: Is it possible to have a dual-boot box with both Solaris 9 and Solaris 10 operating systems on it? If so, how can this be done, and what are the requirements?

A: Yes. You can simply configure the system with two partitions, one with the Solaris 9 OS installed and the other with Solaris 10 OS, and you can switch between them. If you have a system running the Solaris 9 OS today, you can automate this process somewhat with Solaris Live Upgrade. It will allow you to add the Solaris 10 OS (downloaded via Software Express) to your system and rollback to Solaris 9 OS when required. Please see docs.sun.com for more details.

Q: I just installed the Solaris 10 OS on a system (Ultra 5) and issued the DTrace -l command. The system is running extremely slowly now and appears to be cranking away on something, even after Ctrl-c to terminate the DTrace -l call. Did I do something wrong?

A: This is probably a very small memory configuration and a very slow box. On most systems you won't notice the lag. If you run DTrace -l again, just be patient — it will take a while to decompress some in-kernel data, but it will finish.



Sun Expert Exchange

Technical Knowledge Base for Sun Inner Circle Members



Fast Track to Solaris 10 Adoption: Dynamic Tracing Compatibility Issues

1. How does DTrace work with third-party kernel modules like the Veritas VM/VXFS or third-party drivers?
2. Does DTrace work on x86?
3. I've only just begun learning DTrace. Does it have utility in distributed computing environments such as Berkeley Lab's Netlogger?
4. Does DTrace work inside a Solaris virtualization environment?
5. Will any of the standard performance tools, such as SAR, be re-written to make use of DTrace as a back-end data provider?

Q: How does DTrace work with third-party kernel modules like the Veritas VM/VXFS or third-party drivers?

A: The instrumentation methodology that DTrace uses doesn't rely on any modifications to underlying source code. So DTrace can instrument kernel modules that it has never seen before — including Veritas VxVM/VxFS, etc.

Q: Does DTrace work on x86?

A: DTrace is fully functional on both SPARC and x86 platforms. And most of our development (and about 99 percent of our public demos) is on our x86-based Solaris laptops.

Q: I've only just begun learning DTrace. Does it have utility in distributed computing environments such as Berkeley Lab's Netlogger?

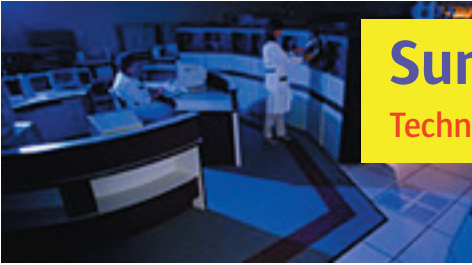
A: NetLogger and DTrace are largely complementary. NetLogger lets you know that you have a problem on a specific machine; DTrace allows you to resolve the root-cause of that problem.

Q: Does DTrace work inside a Solaris virtualization environment?

A: At present, DTrace doesn't work inside of Zones (N1 Grid Containers). However, from the global zone, you can run DTrace to trace processes in any zone or see interactions between zones.

Q: Will any of the standard performance tools, such as SAR, be re-written to make use of DTrace as a back-end data provider?

A: SAR and the standard performance tools are really more performance monitors — they indicate that you have a problem. DTrace allows you to deep-dive to resolve that problem. So they're complementary technologies.



Sun Expert Exchange

Technical Knowledge Base for Sun Inner Circle Members



Fast Track to Solaris 10 Adoption: Dynamic Tracing Functionality & Usability Issues

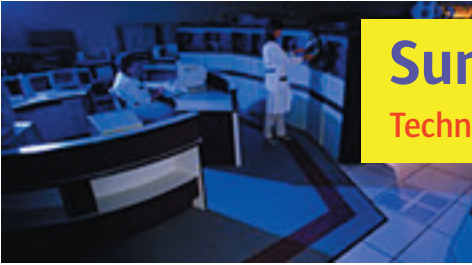
1. How deeply can DTrace peer into applications? Does it see only system calls, or can it see app functions as well?
2. Can DTrace be used to back-trace a core?
3. Does DTrace have all the capabilities that truss has?
4. How much overlap is there, if any, between DTrace and MDB?
5. Can you write your own simulation? If yes, what language would you use to write it?
6. Does Predictive Self-Healing use DTrace or facilities that DTrace also uses?
7. Can DTrace emulate the functionality of iostat/vmstat/mpstat?
8. Can DTrace look inside the Java process called from an application server to monitor specific Java functions?
9. Do I have to understand the kernel to make sense of kernel instrumentation?
10. Are there any plans to provide a GUI wrapper for output, a la SE Toolkit? Or will TCL and kin do the job? If so, are there any special procedures required?
11. What is “anonymous tracing”?
12. Do I need to be root to use DTrace?
13. What level of expertise would I need to use DTrace to solve an application bottleneck problem that I am experiencing?
14. Do you have to know D programming to use DTrace?
15. Can I run multiple copies of DTrace simultaneously? Also can I run DTrace to follow a parent process like truss does?
16. We run an rdist application that consistently crashes our server. No useful messages are generated. Would DTrace be an appropriate tool for tracing this, or is would it be better to use MDB?
17. To diagnose problems, would you recommend running DTrace in a production system?
18. Can DTrace be used in a grid environment — to look at the grid in whole?
19. Can I use DTrace to identify network problems like snoop and tcpdump does?

Q: How deeply can DTrace peer into applications? Does it see only system calls, or can it see app functions as well?

A: You can see system calls and application function calls, as well as every single instruction in your application. For example, you can use DTrace to time how long you spend in function calls, or the exact sequence of instructions that a thread executed as it hit an error condition in a function. And that’s just an example of the flexibility that DTrace offers.

Q: Can DTrace be used to back-trace a core?

A: DTrace is for observing dynamic, running, behavior on the system. For debugging core files, try our post-mortem tool `mdb(1)`. You can also use `pstack(1)` to print the stack backtrace for a live process or core file.



Sun Expert Exchange

Technical Knowledge Base for Sun Inner Circle Members



Q: Does DTrace have all the capabilities that truss has?

A: Almost. DTrace lets you trace all system calls and application function calls (and with much less overhead by the way). It also lets you do much more, including tracing kernel function call, and any instruction in an application (not just function entry and return). Perhaps most usefully, DTrace is much more customizable than truss(1) so you can trace exactly the data you want rather than what truss(1) decides to give you. The one thing truss(1) has that DTrace doesn't is the ability to pretty-print system call arguments.

Q: How much overlap is there, if any, between DTrace and MDB?

A: They're complementary. MDB is for postmortem analysis, while DTrace is for in-situ analysis. Both technologies use some of the same foundation technologies in the Solaris OS, however (for example, both MDB and DTrace use CTF, our format for type information).

Q: Can you write your own simulation? If yes, what language would you use to write it?

A: No. DTrace doesn't offer any simulation ability, and it never allows you alter the system in a way that could be potentially fatal.

Q: Does Predictive Self-Healing use DTrace or facilities that DTrace also uses?

A: Predictive Self-Healing uses a different event channel than DTrace and only shares facilities in that it relies on the robust and full-featured Solaris 10 OS kernel.

Q: Can DTrace emulate the functionality of iostat/vmstat/mpstat?

A: iostat/vmstat/mpstat are monitoring tools. You still want to use those, because they indicate that you have a problem. Then you can use the DTrace io, vminfo and sysinfo providers (respectively) to dive deep. For example, let's say that iostat indicates a bunch of I/O to a device named "sd2." What's causing this? DTrace `-n io start'/args[1]->dev_statname == "sd2"/{@[execname] = count()}'` will do the trick... See the io, vminfo and sysinfo chapters in the AnswerBook for gobs of examples of using iostat/vmstat/mpstat along with DTrace to find and root-cause problems.

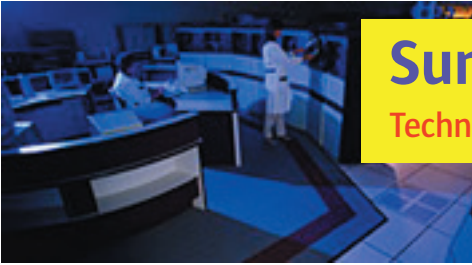
Q: Can DTrace look inside the Java process called from an application server to monitor specific Java functions?

A: Due to the complexities of the Java virtual machine, we can't currently trace Java functions. BUT, you can get Java stack backtraces using the `ustack()` action. Say for example, you want to know the Java activity causing disk I/O. To answer that you'd do something like this
`DTrace -n io start' {@[ustack(50,1000)] = count() }'`

This aggregates based on the call stack and, when you hit ^C, DTrace(1M) will print a table of the stack backtrace and the frequency count. Check out the Solaris Dynamic Tracing Guide for more details on `ustack()`, aggregations and the I/O provider.

Q: Do I have to understand the kernel to make sense of kernel instrumentation?

A: No. We have developed instrumentation with well-defined semantics for CPU scheduling, process control, and I/O (with many more on the way). So you don't need to understand any



Sun Expert Exchange

Technical Knowledge Base for Sun Inner Circle Members



of the kernel implementation to make use of DTrace.

Q: Are there any plans to provide a GUI wrapper for output, a la SE Toolkit? Or will TCL and kin do the job? If so, are there any special procedures required?

A: While one could provide a GUI wrapper, we're actually focused on larger issues. DTrace completely changes the data that you can gather from the system for purposes of visualization. We have some exciting work going on in this department, but it's all very early.

In the more immediate term, we will be providing Java bindings to the libdtrace API — allowing Java programs to act as DTrace consumers. A TK binding should be a snap as well.

Q: What is “anonymous tracing”?

A: Anonymous tracing is a way to use DTrace without a running DTrace(1M) process. This is primarily useful for tracing during boot. See the chapter on anonymous tracing in the AnswerBook guide for details.

Q: Do I need to be root to use DTrace?

A: By default, yes. However, with the new Solaris OS privilege model, you really just need the appropriate DTrace privileges — which can be granted to non-root users.

Q: What level of expertise would I need to use DTrace to solve an application bottleneck problem that I am experiencing?

A: DTrace is one of these tools that's like perl(1): once you know a little, you can start to do quite a bit, and each time you turn to the answer book, you can do even more. We've worked hard on the DTrace answer book to make sure it has that same impact — spending just a few minutes looking at examples will give you the tools to start working on your application problem, and the more you know about DTrace the more you'll be able to do.

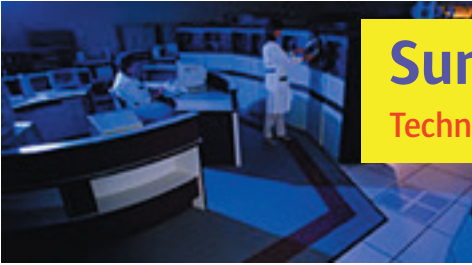
The short answer is that you can get started on solving application problems immediately. Try enabling all the function entry probes in a process (DTrace -n pid entry), and already you're using DTrace to observe your app. Add in a call to the trace() action and you can examine arguments. You'll find that you can start to iterate very quickly on your problem.

Q: Do you have to know D programming to use DTrace?

A: Not really, or at least not at first. For example “DTrace -n xcalls” is a valid use for DTrace that involves absolutely no D, but we think you'll find that D is really easy to get into. To expand on that example, “DTrace -n xcalls'{trace(execname)}” traces the name of every application that induces a cross-call. Technically, this uses D — but “trace(execname)” is pretty simple.

Q: Can I run multiple copies of DTrace simultaneously? Also can I run DTrace to follow a parent process like truss does?

A: You can run as many copies of DTrace(1M) as you like. DTrace(1M) currently doesn't let you follow a process like truss(1) does, but we're actively working on exactly that and hope to make it available in Solaris Express soon.



Sun Expert Exchange

Technical Knowledge Base for Sun Inner Circle Members



Q: We run an rdist application that consistently crashes our server. No useful messages are generated. Would DTrace be an appropriate tool for tracing this, or is would it be better to use MDB?

A: If you mean that Solaris OS crashes, that should never happen. You should get that crash dump to your support person, and they'll use MDB to understand it. If "crash the server" means that an application is crashing, then the answer is still probably to use MDB on the core file, but DTrace may be useful here if it's completely reproducible.

Q: To diagnose problems, would you recommend running DTrace in a production system?

A: Absolutely. Running in production environments is the design center of DTrace, and it (more than anything else) is what separates DTrace from everything that has come before it — for any system. We're so confident about running DTrace in production, that when we last demo'd it to a large group of customers here in Menlo Park, we demo'd it on our local production NFS machine with 1000+ users. So yes, we recommend running DTrace in production.

Q: Can DTrace be used in a grid environment — to look at the grid in whole?

A: DTrace operates on a single system. If you have multiple Zones (N1 Grid Containers), you can use DTrace to observe activity across every zone, including interactions between zones.

Q: Can I use DTrace to identify network problems like snoop and tcpdump does?

A: This is an area that we're actively working on. We're currently developing stable networking providers — networking equivalents of the I/O, proc, sched providers. You should see the first fruits of our labors in the coming months. Stay tuned.